METHOD AND SYSTEM FOR AUTOMATICALLY PROVIDING NETWORK-TRANSACTION-STATUS UPDATES

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] Not applicable.

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

[0002] Not applicable.

TECHNICAL FIELD

[0003] This invention relates to the fields of telecommunications networking and computer programming. More particularly, it relates to processing network transactions and monitoring the progression of the same.

BACKGROUND OF THE INVENTION

[0004] A telecommunications network is composed of a wide array of devices that employ a litany of technologies. A common task involves processing data transactions. Although a transaction may take many forms, typically a transaction updates a database. Databases are ubiquitous throughout a network and may exist in many devices: switches, routers, signal transferors, etc. In some instances, a database is nothing more than a collection of records. But often, a database is quite complex and executing a transaction is commensurately complex. A transaction may be necessary to update routing. Destination addresses may need to be changed. Switch designations may need to be modified. The types and ramifications of the different types of database modifications (transactions) are nearly limitless.

[0005] Executing a transaction may require several substeps or processes to be performed. To analogize, mailing a letter to a destination requires several substeps: collection from a sending location, delivery to sorting facility, emergence from a series of sorting events, distribution to a delivery facility, etc. Continuing this parallel, the prior art does not provide for a way to automatically inform a user that these subprocesses have been completed.

[0006] In the telecommunications environment, a transaction may need to be received, transmitted, scheduled, validated, etc., before finally completing. A problem that plagues communications carriers is related to monitoring the progression of transactions. Historically, a manual, one-transaction-at-a-time query method is used to monitor transaction processing. Such a method is slow and prone to error. To successfully query, a user had to know what information to key in, what information to monitor, what interfaces to navigate and type in search criteria, and how to interpret the data that was returned.

[0007] This prior art method was accomplished by successive query requests manually submitted. One of the easiest ways this was accomplished was for a user to repetitively press an update button, whereby a single transaction's status would be provided. Successive button presses would produce successive updates. But even this method required manual intervention, prevented a user from accomplishing other tasks until the transaction executed, and retrieved updates regarding only a single transaction.

[0008] The prior art relied upon a "pull" type of technology, where a user would manually "pull" or request status updates. Because users would typically be locked out of their workstation as the transaction completed, they would repeatedly submit status-update requests to determine whether the transaction was processing as anticipated and to know when they would

be able to move on to other tasks. These many requests would consume valuable bandwidth and slow related systems.

[0009] This method of manually requesting updates required a user to have in-depth knowledge of a desired transaction's attributes in order to submit an update request. Moreover, the user who submits a request is typically the only user who can monitor a request. The querying process itself slows the notification that the user needs regarding the status of the desired transaction. Transactions are limited to being monitored one at a time in any prior-art schemes.

[0010] The present state of the art could be improved by providing a transaction-processing system that dynamically provides transaction-processing status updates automatically in real time. The art could be further improved by providing a method to asynchronously facilitate the execution of transactions in a networking environment. Finally, there is a need to be able to provide transaction status updates to multiple, rather than a single, receiving component.

SUMMARY OF THE INVENTION

[0011] The present invention is a system that automatically provides unsolicited, real-time status updates of transactions completing in a network environment. The invention has several practical applications in the technical arts, such as providing transaction-status updates without user interaction, thereby freeing resources and eliminating a need for manual update requests. The present invention also enables asynchronous transaction execution, which enables a user to pursue other tasks while a transaction is completing its various substeps. The present invention also provides the ability to broadcast status updates to multiple workstations, screens, and/or persons, including to those other than the transaction initiator. Many more applications and uses will become evident after reading this disclosure.

[0012] Replacing a laborious, time-intensive manual process that relied on a pull type of technology where a user manually submitted successive requests to receive updates regarding a single transaction, the present invention automatically pushes transaction updates to one or more users. Transaction updates may take the form of status indicators that correspond to various stages of a processing transaction as it progresses toward completion. The present invention does not require a user to have a high degree of knowledge of a certain transaction's attributes to receive updates. Rather, those updates can be configured to be automatically sent to the user.

[0013] Reception of updates is not limited to the user who submitted the request. Nodes can be designated to receive updates. The present invention reduces the time a transaction takes to complete by eliminating the bandwidth allocated to receiving the manual transaction-update requests. Multiple transactions can be monitored simultaneously by the present invention.

[0014] In a first aspect, a method is provided for automatically presenting the progress of a transaction. The method includes receiving a transaction to execute. Typically, the transaction will require completing many substeps. Without user interaction, the system communicates to one or more display devices statuses related to the substeps. Statuses such as "begun," "in progress," "executed," "submitted," "validated," "accepted," etc., can be provided. Updates sent from various individual users can be sent to a single node to be monitored rather than merely returned to the submitting user's node.

[0015] In another aspect of the invention, a computer program product is provided that, when executed, provides for automatic, real-time, transaction-progression status updates. Transactions are received and configurable status indicators are generated that correspond to the transaction's progress. Feedback is dynamically communicated to a broadcasting device, which allows status

updates to be sent to multiple receiving components (or aggregated to a single receiving component) without user interaction.

[0016] In still another aspect, a system is provided for monitoring the progression of transactions submitted to a network. The system includes a request-receiving component that receives an incoming transaction; a status-monitoring component that monitors the progression of the transaction and provides feedback related to the status of the transaction's progression toward completion; and a status-transmission component that receives the feedback and communicates the feedback to a group of receiving devices.

[0017] In still another aspect of the invention, a system is provided for asynchronously monitoring network transactions in real time. The system includes a user-interface component for submitting one or more transaction requests, a transaction-processing system for executing transactions, and another user-interface component(s) (which could be the same as the first user-interface component) for receiving the updates and simultaneously presenting them, even if the updates relate to distinct transactions.

[0018] In a final aspect, the present invention includes a method for performing transaction updates asynchronously. The method includes receiving a request to execute one or more transactions. This request typically comes from a user but may come from another electronic device. Even though processing control is withheld from the user during the short time that the transaction is communicated to a receiving component, processing control is returned to the user before the transaction is completely executed, which normally will include completing many substeps.

BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWINGS

[0019] The present invention is described in detail below with reference to the attached drawing figures, wherein:

[0020] FIG. 1A is a block diagram depicting an illustrative operating environment suitable for practicing the present invention;

[0021] FIG. 1B is a component diagram illustrating an embodiment of the present invention having a client component and a server component;

[0022] FIG. 2A is a combined block diagram and flow diagram illustrating an embodiment of the present invention in greater detail;

[0023] FIG. 2B is a flow diagram in accordance with an embodiment of the present invention; and

[0024] FIG. 3 is a flow diagram in accordance with a second embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0025] The present invention automatically provides status updates related to processing a transaction in a communications network environment. Various technical terms are used throughout this description of the present invention. A definition of such terms can be found in Newton's Telecom Dictionary by H. Newton, 19th Edition (2003). These definitions are intended to provide a clearer understanding of the ideas disclosed herein but are in no way intended to limit the scope of the present invention. The definitions and terms should be interpreted broadly and liberally to the extent allowed by the meaning of the words offered in the above-cited reference.

[0026] As one skilled in the art will appreciate, the present invention may be embodied as, among other things a method, system, or computer-program product. Accordingly, the present invention may take the form of a hardware embodiment, a software embodiment, or an embodiment combining software and hardware. In a preferred embodiment, the present invention takes the form of a computer-program product that includes computer-useable instructions embodied on a computer-readable medium.

[0027] Computer-readable media include both volatile and nonvolatile media, removable and nonremovable media, and contemplates media readable by a database, a switch, and various other network devices. Network switches, routers, and related components are conventional in nature, as are the means of communicating with the same. By way of example, and not limitation, computer-readable media include data-storage media and communications media.

[0028] Data-storage media, or machine-readable media, include media implemented in any method or technology for storing information. Examples of stored information include computer-useable instructions, data structures, program modules, and other data representations. Computer-storage media include, but are not limited to RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, Digital Versatile Discs (DVD), holographic media or other optical storage devices, magnetic cassettes, magnetic tape, magnetic disk storage, and other magnetic storage devices. These memory components can store data momentarily, temporarily, and/or permanently.

[0029] Communications media typically store computer-useable instructions – including data structures and program modules – in a modulated data signal. The term "modulated data signal" refers to a propagated signal that has one or more of its characteristics set or changed to encode information in the signal. An exemplary modulated data signal includes a carrier wave or other

transport mechanism. Communications media include any information-delivery media. By way of example but not limitation, communications media include wired media, such as a wired network or direct-wired connection, and wireless media such as acoustic, infrared, radio, microwave, spread-spectrum, and other wireless media technologies. Combinations of the above are included within the scope of computer-readable media.

[0030] FIG. 1A depicts an illustrative operating environment suitable for practicing the present invention. FIG. 1A includes a group of client computers 110 coupled to a server system 112 by a LAN 114. Server system 112 is coupled to a communications network 116, which includes many network elements 118. The invention includes an application 120, which is composed of a client component 120C and server component 120S (see FIG. 1B) in a preferred embodiment. In this embodiment, server component 120S runs on server system 112 and client component 120C runs on client machine 110, which can be a Java-based program that uses a publish/subscribe event-distribution model to provide the status updates to designated user interfaces. Similar to the physical-world, magazine-distribution model, events can be published and those designated to receive them do so.

[0031] The present invention enables transaction requests to be initiated at client machine 110 and monitored as various substeps of the transaction are completed throughout the communications network 116. A common transaction is a database update. A communications network may have several hundred or thousands of databases that describe customer features, call-routing instructions, and countless other data sets. Transactions can take on other forms as well. For instance, a transaction may implement a Local Exchange Routing Guide (LERG) update. The LERG is a document issued by Telcordia Technologies, Inc., of Morristown, NJ (formerly Bellcore) that is used to identify NPA-NXX routing and homing information as well as

network-element and equipment designation. The LERG contains a listing of local-routing data such as destination codes, switching entities, rate centers and locality information by LATA. The LERG is an essential tool for network planning but is in a constant state of flux. LERG updates are well known in the art (albeit monitoring their implementation is resource intensive absent the present invention) and are commonly required on a daily basis.

[0032] As the transactions complete in communications network 116, status updates are automatically returned to one or more client components 120C. A more detailed explanation of an exemplary process is provided with reference to FIG. 2A.

[0033] Turning now to FIG. 2A, a more detailed process for monitoring in real time the progression of transactions is provided. FIG. 2A includes client machine 110, application 120S, event broadcaster 128, request server 129, request table 130, validator 132, universal-command generator 134, transaction sender 136, and responder 138. A component for indicating other processes 140 is also provided.

[0034] The present invention enables a transaction to be received that requires various subprocesses to be performed. Status indicators associated with each of the individual subprocesses (or certain designated processes) are generated and dynamically communicated to a broadcasting device. The broadcasting device sends the status indicators to designated receiving components.

[0035] At a step 210, request server 129 receives a transaction request from client machine 110. In a preferred embodiment, client component 120C is used to create a transaction request. This transaction request requires several subprocesses to complete, which will be described in greater detail below. The present invention enables a user to view the progression of the transaction towards completion. At a step 212, request server 129 adds data corresponding to the

transaction to request table 130. Request table 130 stores the data relating to the transaction request and its progression.

[0036] At a step 214, request server 129 notifies application 120S that the transaction has been requested. This notification can be passed to event broadcaster 128, which automatically communicates the update in real time to designated instances of client application 120C. Request server 129 sends the transaction to validator 132 at a step 216. Validator 132 optionally applies a set of business-validation rules to the transaction to insure that it is in a proper format to be implemented.

[0037] Validator 132 can perform such operations as syntax checking, format checking, transaction-integrity checking, and the like. Validator 132 then places transaction-related data in request table 130 at a step 218. At a step 220, validator 132 sends the transaction data to both the universal command generator 134 and transaction sender 136. Transaction sender 136 sends the transaction to the respective network elements that are necessary to process the transaction.

[0038] Having completed another exemplary subprocess of the transaction, validator 132 sends a message to application 120S that a transaction is ready to be sent to the various network elements. Application 120S then communicates this event-to-event broadcaster 128, which can dynamically communicate this message to the various client machines 110. At a step 224, transaction sender 136 sends the transaction to the various network elements that are necessary to process the transaction. At a step 226, transaction sender 136 receives the response or responses from network elements 118. Responder 138 then updates requests table 130 at a step 228. Responder 138 also sends a message to application 120S that the transaction is complete, that it is successfully processed by network elements 118 at a step 230. Again application 120S communicates this update to event broadcaster 128, which immediately communicates this status

update to various client devices 110. Instead of sending status updates to end-user devices, event broadcaster 128 can also provide input to other processes 140.

[0039] Turning now to FIG. 3, an alternative embodiment for practicing the present invention is provided. Not all steps in FIG. 3 are required steps and some steps may be performed in a different order. At a step 310 a transaction request is originated. This transaction request can be originated by client device 110 using client component 120C. As previously mentioned, client component 120C includes a user interface for transmitting transaction requests and receiving transaction updates. In alternative embodiments, a second user interface can be used to receive status updates. In some embodiments status updates are sent to multiple client devices 110, and in other embodiments all updates can be sent to a single client device 110.

[0040] In this later embodiment, a single person can monitor all updates and respond accordingly. Such an arrangement allows increased efficiencies by reducing the amount of resources necessary to monitor and address transaction-update progression. The transaction is submitted at a step 312 and optionally validated at a step 314. The transaction can be accepted by a variety of devices including a processing system such as system 234 (see FIG. 2A).

[0041] Different network elements such as communication switches, routers, etc., may require the transaction to be formatted. Accordingly, the transaction can be formatted as required at a step 318. This transaction is then sent to various network elements 118 at a step 320. Responses are received from the various devices indicating whether the transaction was successful or not at a step 322. Appropriate messages are sent to the different client-user interfaces 120C based on the receiver responses at a step 324.

[0042] A status update can be sent to various client devices based on any of the aforementioned steps (210-230 or 310-324). In this manner, the present invention receives a

request to execute potentially multiple transactions. Instead of withholding process control from a user until the entire transaction is processed, processing may be withheld only until such time that a receiving device (such as request server 129 or processing system 234) receives the transaction request. Whereas in prior-art systems, processing was delayed until the entire transaction had completed, the present invention allows user control to be returned more quickly, as soon as a receiving device receives the transaction request. Control is returned to a user incident to receiving a transaction request but prior to actually executing the entire transaction request at the various network elements 118. The updates are sent automatically without a user having to manually request status updates.

[0043] As can be seen, the present invention and its equivalents are well adapted to provide a new and useful method for, among other things, providing status updates associated with the progression of a transaction. Many different arrangements of the various components depicted, as well as components not shown, are possible without departing from the spirit and scope of the present invention.

[0044] The present invention has been described in relation to particular embodiments, which are intended in all respects to be illustrative rather than restrictive. Alternative embodiments will become apparent to those skilled in the art that do not depart from its scope. Many alternative embodiments exist but are not included because of the nature of this invention. A skilled programmer may develop alternative means of implementing the aforementioned improvements without departing from the scope of the present invention.

[0045] It will be understood that certain features and subcombinations are of utility and may be employed without reference to other features and subcombinations and are contemplated

within the scope of the claims. Not all steps listed in the various figures need to be carried out in the specific order described.